

# Objetivos

Aprender a manejar JDBC mediante una pequeña aplicación de gestión de coches

## Requerimiento 1

Se desea hacer un CRUD completo de la entidad 'Coche', pero esta vez no se trabajará con ningún fichero, se trabajará con una BBDD. Es muy importante usar el patrón DAO visto en clase. Los parámetros de conexión a la BBDD deben estar hechos en un fichero de **propiedades**.

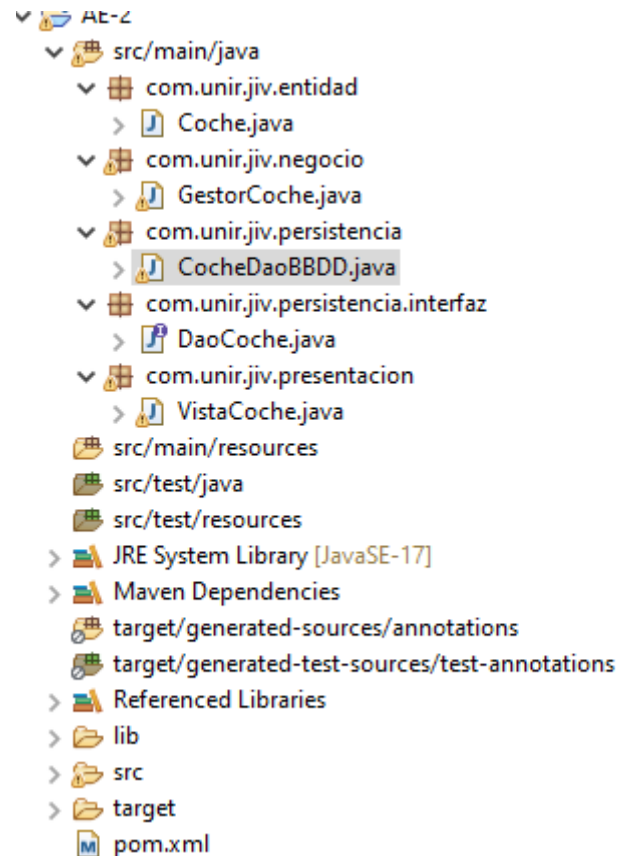
El coche tendrá los siguientes atributos: id, marca, modelo, año de fabricación y km.

El menú mostrado será de la siguiente forma:

- Añadir nuevo coche (El ID lo incrementará automáticamente la base de datos)
- Borrar coche por ID
- Consulta coche por ID
- Modificar coche por ID (pedirá todos los valores y modificará dichos valores a partir del ID del coche)
- Listado de coches
- Terminar el programa

Valoración: 4 puntos sobre 10

## Estructura del proyecto



Conexión a la base de datos:

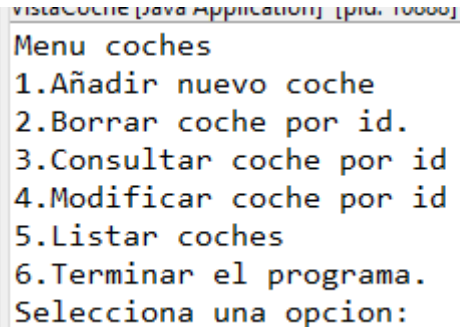
```
15
16 public class CocheDaoBBDD implements DaoCoche{
17
18     private Connection conexion;
19     private static Scanner leer = new Scanner(System.in);
20     public CocheDaoBBDD() {
21         abrirConexion();
22     }
23
24     @Override
25     public boolean abrirConexion(){
26         String url = "jdbc:mysql://localhost:3306/gestion_coches";
27         String usuario = "root";
28         String password = "";
29         try {
30             conexion = DriverManager.getConnection(url,usuario,password);
31         } catch (SQLException e) {
32             // TODO Auto-generated catch block
33             e.printStackTrace();
34             return false;
35         }
36         return true;
37     }
}
```

## Menú escrito en la clase VistaCoche

```
do {
    System.out.println("Menu coches");
    System.out.println("1.Añadir nuevo coche");
    System.out.println("2.Borrar coche por id.");
    System.out.println("3.Consultar coche por id");
    System.out.println("4.Modificar coche por id");
    System.out.println("5.Listar coches");
    System.out.println("6.Terminar el programa.");
    System.out.println("Selecciona una opcion:");
    opcion = sc.nextInt();

    switch(opcion) {
        case 1:
            cd.addCoche();
            break;
        case 2:
            System.out.println("Introduce el ID del coche a borrar:");
            int id = sc.nextInt();
            cd.borrarCoche(id);
            break;
        case 3:
            System.out.println("Introduce el ID del coche a consultar:");
            int idConsultar = sc.nextInt();
            cd.getCochePorId(idConsultar);
            break;
        case 4:
            System.out.println("Introduce el ID del coche a modificar:");
            int idModificar = sc.nextInt();
            cd.editarCoche(idModificar);
            break;
        case 5:
            cd.listarCoches();
            break;
        case 6:
            System.out.println("Fin del programa");
            break;
        default:
            System.out.println("Opcion no valida.");
            System.out.println("Introduzca un numero del 1 al 5, para terminar elija 6.");
    }
}while(opcion != 6);
```

## Representación en consola



```
VistaCoche Java Application [pid: 10000]
Menu coches
1.Añadir nuevo coche
2.Borrar coche por id.
3.Consultar coche por id
4.Modificar coche por id
5.Listar coches
6.Terminar el programa.
Selecciona una opcion:
```

## Listado de coches

```
Selecciona una opcion:
5
Coche [id=3, marca=Seat, modelo=arosa, anio=2020, km=12332.0]
Coche [id=4, marca=ford, modelo=focus, anio=2010, km=82322.0]
Menu coches
1.Añadir nuevo coche
2.Borrar coche por id.
3.Consultar coche por id
4.Modificar coche por id
5.Listar coches
6.Terminar el programa.
Selecciona una opcion:
```

Código para borrar coches de la BBDD:

```
@Override
public List<Coche> listarCoches() {
    String query = "select * from coches";
    List<Coche> lista = new ArrayList<>();
    try {
        PreparedStatement ps = conexion.prepareStatement(query);
        ResultSet rs = ps.executeQuery();
        while(rs.next()) {
            Coche coche = new Coche();
            coche.setId(rs.getInt("id"));
            coche.setMarca(rs.getString("marca"));
            coche.setModelo(rs.getString("modelo"));
            coche.setAnio(rs.getInt("anio"));
            coche.setKm(rs.getDouble("km"));
            lista.add(coche);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    for(Coche ele:lista) {
        System.out.println(ele);
    }
    return lista;
}

@Override
```

## Modificar coches. Código y representación consola

```
109  @Override
110  public Coche editarCoche(int id) {
111      Coche coche = null;
112      Scanner leer = new Scanner(System.in);
113      System.out.println("Ingresar nombre de marca:");
114      String marca = leer.next();
115      System.out.println("Ingresar modelo:");
116      String modelo = leer.next();
117      System.out.println("Introducir año");
118      int anio = leer.nextInt();
119      System.out.println("Introducir numero de kilometros");
120      double km = leer.nextDouble();
121
122      String query = "update coches set marca=?,modelo=?,anio=?,km=?"
123          + " where id=?";
124      try {
125          PreparedStatement ps = conexion.prepareStatement(query);
126          ps.setString(1, marca);
127          ps.setString(2, modelo);
128          ps.setInt(3, anio);
129          ps.setDouble(4, km);
130          ps.setInt(5, id);
131          ps.executeUpdate();
132      } catch (SQLException e) {
133          e.printStackTrace();
134      }
135      return coche;
136  }
```

Problems Javadoc Declaration Console X

VistaCoche [Java Application] [pid: 10888]

5. Listar coches

6. Terminar el programa.

Selecciona una opcion:

4

Introduce el ID del coche a modificar:

4

Ingresar nombre de marca:

ford

Ingresar modelo:

focus

Introducir año

1999

Introducir numero de kilometros

2132

## Eliminar coche

```
@Override
public Coche borrarCoche(int id) {
    if(!abrirConexion()) {
        return null;
    }

    String query = "delete from coches where id =?";
    try {
        PreparedStatement ps = conexion.prepareStatement(query);
        ps.setInt(1, id);
        ps.executeUpdate();
        System.out.println("Coche eliminado");
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return null;
}
```

## Resultado en consola

```
Coche [id=3, marca=Seat, modelo=arosa, anio=2020, km=12332.0]
Coche [id=4, marca=ford, modelo=focus, anio=1999, km=2132.0]
Menu coches
1.Añadir nuevo coche
2.Borrar coche por id.
3.Consultar coche por id
4.Modificar coche por id
5.Listar coches
6.Terminar el programa.
Selecciona una opcion:
2
Introduce el ID del coche a borrar:
4
Coche eliminado
Menu coches
1.Añadir nuevo coche
2.Borrar coche por id.
3.Consultar coche por id
4.Modificar coche por id
5.Listar coches
6.Terminar el programa.
Selecciona una opcion:
5
Coche [id=3, marca=Seat, modelo=arosa, anio=2020, km=12332.0]
Menu coches
1.Añadir nuevo coche
```

## Requerimiento 2

Se pide añadir la siguiente funcionalidad.

Los coches, tendrán asociados N pasajeros en él (habrá que crear la tabla pasajeros y hacer la relación pertinente). Los pasajeros tendrán los siguientes atributos, id, nombre, edad y peso. Se añadirá la opción “gestión de pasajeros” al programa principal, dicha opción nos mostrará un submenú como el que sigue

- Crear nuevo pasajero
- Borrar pasajero por id
- Consulta pasajero por id
- Listar todos los pasajeros
- Añadir pasajero a coche, el programa nos pedirá un id de un pasajero y el id de un coche, y lo añadirá al coche a nivel de base de datos. Sería una buena opción mostrar todos los coches disponibles.
- Eliminar pasajero de un coche, el programa nos pedirá un id de un pasajero y lo eliminará del coche a nivel de base de datos. Sería una buena opción mostrar todos los coches y sus pasajeros asociados.
- Listar todos los pasajeros de un coche, el programa pedirá el id de un coche, y nos mostrará todos los pasajeros asociados a él.

Menu para gestionar los pasajeros

```
}  
  
private static void gestionPasajeros() {  
    Scanner sc = new Scanner(System.in);  
    int opcion = 0;  
  
    do {  
        System.out.println("*****");  
        System.out.println("Menu coches");  
        System.out.println("1.Crear nuevo pasajero");  
        System.out.println("2.Borrar pasajero por id.");  
        System.out.println("3.Consultar pasajero por id");  
        System.out.println("4.Listar pasajeros");  
        System.out.println("5.Añadir pasajero a coche");  
        System.out.println("6.Eliminar pasajero de un coche.");  
        System.out.println("7.Listar todos los pasajeros de un coche.");  
        System.out.println("8.Salir");  
        System.out.println("Selecciona una opcion:");  
        opcion = sc.nextInt();  
    }  
}
```

Opciones del submenu pasajeros:

```
switch(opcion) {  
case 1:  
    pd.nuevoPasajero();  
    break;  
case 2:  
    System.out.println("Introduce el ID del pasajero a borrar:");  
    int id = sc.nextInt();  
    pd.borrarPasajero(id);  
    break;  
case 3:  
    System.out.println("Introduce el ID del pasajero a buscar:");  
    int idPasajero = sc.nextInt();  
    pd.consultarPasajero(idPasajero);  
    break;  
case 4:  
    pd.listarPasajeros();  
    break;  
case 5:  
    System.out.println("Introduce id del pasajero a añadir");  
    int idPersona = sc.nextInt();  
    System.out.println("Introduce id del coche");  
    int idCoche = sc.nextInt();  
    pd.addPasajeroACoche(idPersona, idCoche);  
    break;  
case 6:  
    System.out.println("Introduce la id del pasajero a borrar");  
    int idPasajeroBorrar = sc.nextInt();  
    pd.eliminarPasajeroCoche(idPasajeroBorrar);  
    break;  
case 7:  
    System.out.println("Introduce id del coche");  
    int idCocheListar = sc.nextInt();  
    pd.listarPasajerosCoche(idCocheListar);  
    break;  
case 8:  
    System.out.println("Salir del menu pasajero");  
    break;  
default:  
    System.out.println("Opcion no valida.");  
    System.out.println("Introduzca un numero del 1 al 8.");  
}
```

---



Salir del menu pasajeros y volver al menu principal:

```
Selecciona una opcion:
6
*****
Menu pasajeros
1.Crear nuevo pasajero
2.Borrar pasajero por id.
3.Consultar pasajero por id
4.Listar pasajeros
5.Añadir pasajero a coche
6.Eliminar pasajero de un coche.
7.Listar todos los pasajeros de un coche.
8.Salir
Selecciona una opcion:
8
Salir del menu pasajero
-----
Menu coches
1.Añadir nuevo coche
2.Borrar coche por id.
3.Consultar coche por id
4.Modificar coche por id
5.Listar coches
6.Gestionar pasajeros.
7.Terminar el programa.
Selecciona una opcion:
```

Crear nuevo pasajero me da error

```
8
Introducir peso de pasajero:
1
Exception in thread "main" java.lang.IllegalArgumentException: Object: Pasajero [id=0, nombre=Rafael, edad=38, peso=81.0] is not a known Entity type.
    at org.eclipse.persistence.internal.sessions.UnitOfWorkImpl.mergeCloneWithReferences(UnitOfWorkImpl.java:3605)
    at org.eclipse.persistence.internal.sessions.RepeatableWriteUnitOfWork.mergeCloneWithReferences(RepeatableWriteUnitOfWork.java:389)
    at org.eclipse.persistence.internal.sessions.UnitOfWorkImpl.mergeCloneWithReferences(UnitOfWorkImpl.java:3576)
    at org.eclipse.persistence.internal.jpa.EntityManagerImpl.mergeInternal(EntityManagerImpl.java:648)
    at org.eclipse.persistence.internal.jpa.EntityManagerImpl.merge(EntityManagerImpl.java:625)
    at com.unir.jiv.persistencia.PasajeroDao.nuevoPasajero(PasajeroDao.java:81)
    at com.unir.jiv.presentacion.VistaCoche.gestionPasajeros(VistaCoche.java:101)
    at com.unir.jiv.presentacion.VistaCoche.mostrarMenu(VistaCoche.java:65)
    at com.unir.jiv.presentacion.VistaCoche.main(VistaCoche.java:20)
```

Aunque luego en la base de datos se han guardado los datos introducidos.

The screenshot shows the phpMyAdmin interface. On the left is the database navigation tree with 'gestion\_coches' expanded and 'pasajeros' selected. The main panel shows the 'Examinar' (Browse) view of the 'pasajeros' table. A green status bar at the top indicates 'Mostrando filas 0 - 3 (total de 4, La consulta tardó 0,0003 segundos.)'. Below this is the SQL query: `SELECT * FROM `pasajeros``. A toolbar contains options like 'Perfilando', 'Editar en línea', 'Editar', 'Explicar SQL', 'Crear código PHP', and 'Actualizar'. Below the toolbar are filters for 'Mostrar todo', 'Número de filas' (set to 25), 'Filtrar filas' (with a search box), and 'Ordenar según la cl'. An 'Opciones extra' button is also present. The table data is displayed in a grid with columns: id, nombre, edad, peso, and id\_coche. Each row has 'Editar', 'Copiar', and 'Borrar' icons. At the bottom, there is a 'Seleccionar todo' checkbox and a summary for the selected elements: 'Para los elementos que están marcados: Editar, Copiar, Borrar'.

	id	nombre	edad	peso	id_coche
<input type="checkbox"/>	1	Ana	23	49.00	NULL
<input type="checkbox"/>	2	Ana	23	38.00	NULL
<input type="checkbox"/>	3	Eva	38	58.00	NULL
<input type="checkbox"/>	4	Rafael	38	81.00	NULL

Consulta de pasajero por id, muestra errores pero entre las líneas muestra la información solicitada como se ve en la línea subrayada en verde.

```
*****
Menu pasajeros
1.Crear nuevo pasajero
2.Borrar pasajero por id.
3.Consultar pasajero por id
4.Listar pasajeros
5.Añadir pasajero a coche
6.Eliminar pasajero de un coche.
7.Listar todos los pasajeros de un coche.
8.Salir
Selecciona una opcion:
3
Introduce el ID del pasajero a buscar:
1
[EL Severe]: metadata: 2024-02-11 17:46:18.376--ServerSession(1898325501)--The com.unir.jiv.entidad.Pasajero class was compiled with an unsupported JDK. Report this e
java.lang.IllegalArgumentException: Unsupported class file major version 61
[EL Severe]: metadata: 2024-02-11 17:46:18.38--ServerSession(1898325501)--The com.unir.jiv.entidad.Pasajero class was compiled with an unsupported JDK. Report this e
java.lang.ArrayIndexOutOfBoundsException: Index 8 out of bounds for length 0
[EL Info]: 2024-02-11 17:46:18.39--ServerSession(1898325501)--EclipseLink, version: Eclipse Persistence Services - 2.7.7.v20200504-69f2c2b80d
[EL Warning]: metamodel: 2024-02-11 17:46:18.419--The collection of metamodel types is empty. Model classes may not have been found during entity search for Java SE
Exception in thread "main" java.lang.IllegalArgumentException: Object: Pasajero [id=1, nombre=Ana, edad=23, peso=49.0] is not a known Entity type.
    at org.eclipse.persistence.internal.sessions.UnitOfWorkImpl.mergeCloneWithReferences(UnitOfWorkImpl.java:3605)
    at org.eclipse.persistence.internal.sessions.RepeatableWriteUnitOfWork.mergeCloneWithReferences(RepeatableWriteUnitOfWork.java:389)
    at org.eclipse.persistence.internal.sessions.UnitOfWorkImpl.mergeCloneWithReferences(UnitOfWorkImpl.java:3576)
    at org.eclipse.persistence.internal.jpa.EntityManagerImpl.mergeInternal(EntityManagerImpl.java:648)
    at org.eclipse.persistence.internal.jpa.EntityManagerImpl.merge(EntityManagerImpl.java:625)
    at com.unir.jiv.persistencia.PasajeroDao.consultarPasajero(PasajeroDao.java:119)
    at com.unir.jiv.presentacion.VistaCoche.gestionPasajeros(VistaCoche.java:111)
    at com.unir.jiv.presentacion.VistaCoche.mostrarMenu(VistaCoche.java:65)
    at com.unir.jiv.presentacion.VistaCoche.main(VistaCoche.java:20)
```

Dirección de github donde se aloja el proyecto de Jose Ignacio:

<https://github.com/Jose-ignacio-vazquez/AccesoDatos2>